Università di Genova

Corso di Laurea in Ingegneria Informatica

Appunti di Reti di Calcolatori

Prof. Massimo Maresca

a.a. 2019-2020

ISBN: 979-12-200-5974-9

1. Appunto sulla Velocità di propagazione delle onde elettromagnetiche

In questo paragrafo si ricava la velocita' di propagazione delle onde elettromagnetiche dalle Equazioni di Maxwell. L'obiettivo e' quello di rilevare che la velocita' di propagazione e' una grandezza autonoma, indipendente dal bit-rate effettivo, che viene preso in considerazione successivamente.

Dalle equazioni di Maxwell:

1. $\nabla \times \underline{E} = -\frac{\partial \underline{B}}{\partial t}$
2. $\nabla \times \underline{B} = \mu\varepsilon \frac{\partial \underline{E}}{\partial t}$, in assenza di corrente
3. $\nabla \bullet E = 0$, in assenza di cariche.

Applicando l'operatore $\nabla \times$ alla 1., si ha:

$\nabla \times (\nabla \times \underline{E}) = -\nabla \times \frac{\partial \underline{B}}{\partial t}$, da cui si ha: $\nabla \times (\nabla \times \underline{E}) = -\frac{\partial (\nabla \times \underline{B})}{\partial t}$ e quindi:

$\nabla \times (\nabla \times \underline{E}) = -\mu\varepsilon \frac{\partial^2 \underline{E}}{\partial t^2}$.

Consideriamo poi che, dato un generico vettore $\underline{V}$, si ha: $\nabla \times (\nabla \times \underline{V}) = \nabla(\nabla \bullet \underline{V}) - \nabla^2 \underline{V}$, come dimostrato nel seguito:

Calcoliamo dapprima $\nabla \times (\nabla \times \underline{V})$:

$$\nabla \times \underline{V} = \begin{vmatrix} \underline{i} & \underline{j} & \underline{k} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ V_x & V_y & V_z \end{vmatrix} = \underline{i}\left(\frac{\partial V_z}{\partial y} - \frac{\partial V_y}{\partial z}\right) + \underline{j}\left(\frac{\partial V_x}{\partial z} - \frac{\partial V_z}{\partial x}\right) + \underline{k}\left(\frac{\partial V_y}{\partial x} - \frac{\partial V_x}{\partial y}\right)$$

$$\nabla \times (\nabla \times \underline{V}) = \begin{vmatrix} \underline{i} & \underline{j} & \underline{k} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ \frac{\partial V_z}{\partial y} - \frac{\partial V_y}{\partial z} & \frac{\partial V_x}{\partial z} - \frac{\partial V_z}{\partial x} & \frac{\partial V_y}{\partial x} - \frac{\partial V_x}{\partial y} \end{vmatrix} =$$

$$= \underline{i}\left(\frac{\partial^2 V_y}{\partial y \partial x} - \frac{\partial^2 V_x}{\partial y^2} - \frac{\partial^2 V_x}{\partial z^2} + \frac{\partial^2 V_z}{\partial z \partial x}\right) + \underline{j}\left(\frac{\partial^2 V_z}{\partial z \partial y} - \frac{\partial^2 V_y}{\partial z^2} - \frac{\partial^2 V_y}{\partial x^2} + \frac{\partial^2 V_x}{\partial x \partial y}\right) + \underline{k}\left(\frac{\partial^2 V_x}{\partial x \partial z} - \frac{\partial^2 V_z}{\partial x^2} - \frac{\partial^2 V_z}{\partial y^2} + \frac{\partial^2 V_y}{\partial z \partial y}\right)$$

Calcoliamo ora $\nabla(\nabla \bullet \underline{V})$:

$$\nabla \bullet \underline{V} = \frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} + \frac{\partial V_z}{\partial z}$$

$$\nabla(\nabla \bullet \underline{V}) = \underline{i}(\frac{\partial^2 V_x}{\partial x^2} + \frac{\partial^2 V_y}{\partial x \partial y} + \frac{\partial^2 V_z}{\partial x \partial z}) + \underline{j}(\frac{\partial^2 V_x}{\partial y \partial x} + \frac{\partial^2 V_y}{\partial y^2} + \frac{\partial^2 V_z}{\partial y \partial z}) + \underline{k}(\frac{\partial^2 V_x}{\partial z \partial x} + \frac{\partial^2 V_y}{\partial y \partial z} + \frac{\partial^2 V_z}{\partial z^2})$$

$$\nabla^2 \underline{V} = \underline{i}(\frac{\partial^2 V_x}{\partial x^2} + \frac{\partial^2 V_x}{\partial y^2} + \frac{\partial^2 V_x}{\partial z^2}) + \underline{j}(\frac{\partial^2 V_y}{\partial x^2} + \frac{\partial^2 V_y}{\partial y^2} + \frac{\partial^2 V_y}{\partial z^2}) + \underline{k}(\frac{\partial^2 V_z}{\partial x^2} + \frac{\partial^2 V_z}{\partial y^2} + \frac{\partial^2 V_z}{\partial z^2})$$

$$\nabla(\nabla \bullet \underline{V}) - \nabla^2 \underline{V} = \underline{i}(\frac{\partial^2 V_y}{\partial x \partial y} + \frac{\partial^2 V_z}{\partial x \partial z} - \frac{\partial^2 V_x}{\partial y^2} - \frac{\partial^2 V_x}{\partial z^2}) +$$

$$+ \underline{j}(\frac{\partial^2 V_x}{\partial y \partial x} + \frac{\partial^2 V_z}{\partial y \partial z} - \frac{\partial^2 V_y}{\partial x^2} - \frac{\partial^2 V_y}{\partial z^2}) +$$

$$+ \underline{k}(\frac{\partial^2 V_x}{\partial z \partial x} + \frac{\partial^2 V_y}{\partial y \partial z} - \frac{\partial^2 V_z}{\partial x^2} - \frac{\partial^2 V_z}{\partial y^2})$$

Questo dimostra che $\nabla \times (\nabla \times \underline{V}) = \nabla(\nabla \bullet \underline{V}) - \nabla^2 \underline{V}$

Nel caso specifico: $\nabla \times (\nabla \times \underline{E}) = \nabla(\nabla \bullet \underline{E}) - \nabla^2 \underline{E}$

Considerando la 3. si ha $\nabla \times (\nabla \times \underline{E}) = -\nabla^2 \underline{E}$ e quindi $-\nabla^2 \underline{E} = -\mu\varepsilon \frac{\partial^2 \underline{E}}{\partial t^2}$

Nella dimensione x, ad esempio, si ha $\frac{\partial^2 s}{\partial x^2} = \mu\varepsilon \frac{\partial^2 s}{\partial t^2}$ (indicando con s una singola componente), che è la tipica equazione dell'onda.

La soluzione dell'eq. dell'onda è

$s(x,t) = A\cos(wt - kx)$, con $w = 2\pi f$ e $k = 2\pi\lambda$, essendo f la frequenza e λ è la lunghezza d'onda.

Esaminando la progressione del fronte d'onda si ha $wt - kx = 0$, da cui $x = \frac{w}{k}t$, il che implica che la velocità di propagazione è $v = \frac{w}{k}$.

Identifichiamo i parametri sull'equazione dell'onda:

$\frac{\partial s}{\partial x} = Ak\sin(wt - kx)$, $\frac{\partial^2 s}{\partial x^2} = -Ak^2\cos(wt - kx)$

$\frac{\partial s}{\partial t} = -Aw\sin(wt - kx)$, $\frac{\partial^2 s}{\partial t^2} = -Aw^2\cos(wt - kx)$

$Ak^2\cos(wt - kx) = \mu\varepsilon Aw^2\cos(wt - kx)$, da cui si ha $k^2 = \mu\varepsilon w^2$ e di conseguenza

$\frac{w^2}{k^2} = \frac{1}{\mu\varepsilon}$, e quindi $v = \frac{w}{k} = \frac{1}{\sqrt{\mu\varepsilon}}$.

Vediamo la velocità di propagazione nel vuoto partendo dalle due costanti ε_0 e μ_0.

ε_0 = 8.854187817620... × 10^{-12} F·m^{-1}; μ_0 = 4π×10^{-7} V·s/(A·m).
Utilizzando tali valori si ha:

$$v = \frac{1}{\sqrt{8.8 \times 4\pi \times 10^{-19}}} = \frac{1}{\sqrt{11.58 \times 10^{-18}}} = 3 \times 10^{8}$$, che è la velocità della luce.

2. Appunto su Signal Noise Ratio

In questo paragrafo si prende in esame il problema della trasmissione digitale in presenza di rumore. Il problema e' quello di allocare l'estensione dell'ampiezza al massimo numero di bit possibile, compatibilmente con il rumore massimo prevedibile.

Segnale trasmesso S_T: $-S_{MAX} \le S_T \le S_{MAX}$

Noise N: $-N_{MAX} \le N \le N_{MAX}$

Segnale ricevuto S_R: $S_T - N \le S_R \le S_T + N$

Se $S=N$ possiamo supporre di avere due casi:

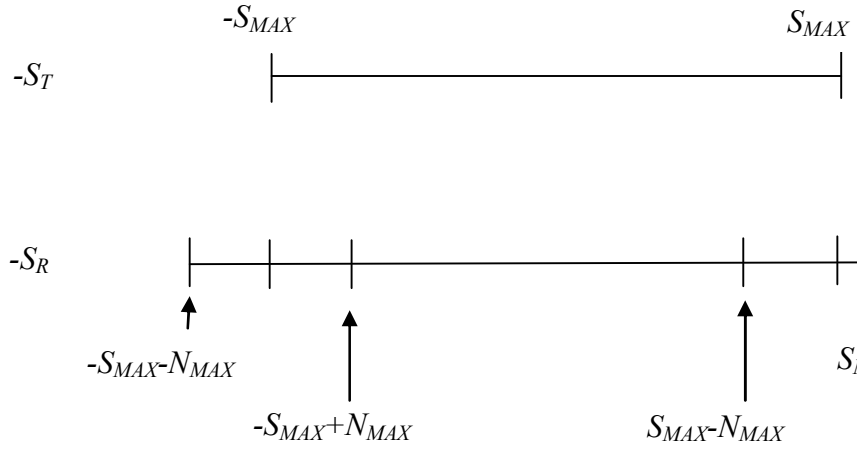
$S_T = -S_{MAX}$: $-S_{MAX} - N \le S_R \le -S_{MAX} + N$ $\quad -2S_{MAX} < S_R < 0$

$S_T = S_{MAX}$: $S_{MAX} - N \le S_R \le S_{MAX} + N$ $\quad 0 < S_R < 2S_{MAX}$

è possibile riconoscere 1 bit codificandolo ad esempio come segue:

$(0 \Leftrightarrow -S_{MAX}, \Leftrightarrow S_{MAX})$

Invece nel caso generale in cui $S_{MAX} >> N_{MAX}$:



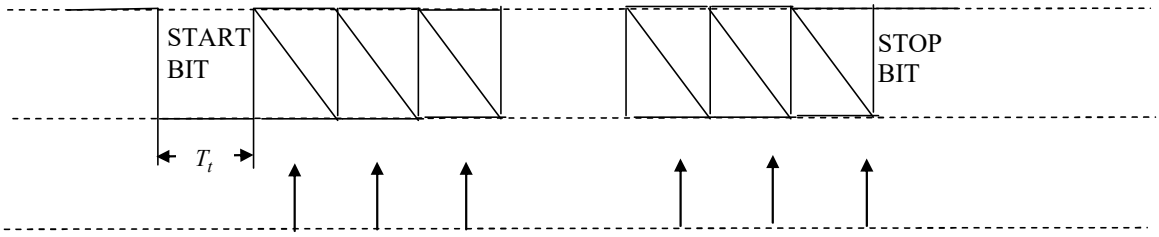
Numero di slot = $\frac{2S_{MAX}}{2N_{MAX}} = \frac{S_{MAX}}{N_{MAX}}$

Numero di bit = $\log_2(1 + \frac{S_{MAX}}{N_{MAX}})$

Rate = $B_T \log_2(1 + \frac{S_{MAX}}{N_{MAX}})$

3. Appunto sulla Trasmissione Seriale Asincrona

In questo paragrafo si prende in esame il problema della sincronizzazione in presenza di trasmissione seriale asincrona. Il trasmettitore ed il ricevitore conoscono il clock della trasmissione con un margine di errore. La sincronizzazione inizia con uno Start Bit e poi procede autonomamente. I singoli bit vengono trasmessi da trasmettitore con il clock di trasmissione e acquisiti dal ricevitore con il clock di ricezione. Cio' che va evitato e' che un bit venga saltato o acquisito due volte.




```
status=IDLE;
switch_on_event(UPTODOWN);
while (TRUE){
   event=wait_event();
   switch (status) {
         case IDLE:       if (event == UPTODOWN){
                                switch_off_event(UPTODOWN);
                                status = RECEIVING;
                                nbit=0;
                                t=gettime()+FULLBITTIME*1,5;
                                set_alarm(t);
                                switch_on_event(ALARM);
                          }
                          break;
         case RECEIVING:  if (event == ALARM) {
                                sequence[nbit]=getinput();
                                ++nbit;
                                if (nbit<N) {
                                      t=gettime()+FULLBITTIME;
                                      set_alarm(t);
                                } else {
                                      consume(sequence);
                                      status = IDLE;
                                      switch_on_event(UPTODOWN);
                                }
                          }
                          break;
   }
}
```

$T_r \bullet 0{,}5 + n \bullet T_r$: Campionamento dell'n-esimo bit in ricezione

$n \bullet T_t - (n+1) \bullet T_t$: Periodo di validità dell'n-esimo bit in trasmissione

Se $T_r > T_t$ per poter ricevere n bit correttamente si deve avere: $T_r \bullet 0{,}5 + n \bullet T_r < (n+1)T_t$

Se $T_r < T_t$ per poter ricevere n bit correttamente si deve avere: $T_r \bullet 0{,}5 + n \bullet T_r > n \bullet T_t$

Problema n. 1:

Dato n = numero di bit effettivi da trasmettere tra due sincronizzazioni, trovare la relazione tra i due clock.

Soluzione:

$$nT_t < T_r(n+0{,}5) < T_t(n+1)$$

$$\frac{n}{n+0{,}5} < \frac{T_r}{T_t} < \frac{n+1}{n+0{,}5}$$

$$\frac{n}{n+0{,}5} < \frac{f_t}{f_r} < \frac{n+1}{n+0{,}5}$$

Supponiamo ora che le frequenze in bps vengano date con una certa precisione e cioè:

$$f_{reale} = f_{nominale} \pm k\%$$

Si ha quindi:

$$f_n(1-k) < f_t < f_n(1+k)$$

$$f_n(1-k) < f_r < f_n(1+k)$$

da cui:

$$\frac{f_t^{MIN}}{f_r^{MAX}} < \frac{f_t}{f_r} < \frac{f_t^{MAX}}{f_r^{MIN}}$$

e

$$\frac{1-k}{1+k} < \frac{f_t}{f_r} < \frac{1+k}{1-k}$$

Di conseguenza per ricevere correttamente n bit si deve avere:

$$\left(\frac{f_t}{f_r}\right)^{MAX} = \frac{1+k}{1-k} < \frac{n+1}{n+0{,}5}$$

$$\left(\frac{f_t}{f_r}\right)^{MIN} = \frac{1-k}{1+k} > \frac{n}{n+0{,}5}$$

Vediamo i due casi:

Caso 1: $\frac{1+k}{1-k} < \frac{n+1}{n+0{,}5}$

Si ha $(1+k)(n+0{,}5) < (1-k)(n+1)$ e con semplici passaggi:

$$n + kn + 0{,}5 + 0{,}5k < n + 1 + 0{,}5$$

$$2kn + 0{,}5k + k < 1 - 0{,}5$$

$$2Kn + 1{,}5k < 0{,}5$$

$$k < \frac{0,5}{2n+1,5}$$

Caso 2: $\frac{1 \quad k}{1+k} > \frac{n}{n+0,5}$

Si ha $2kn + 0,5k < 0,5$ e quindi $k < \frac{0,5}{2n+0,5}$.

Poiché devono verificarsi simultaneamente $k < \frac{0,5}{2n+1,5}$ e $k < \frac{0,5}{2n+0,5}$, e la prima è inclusa nella seconda si ha: $k < \frac{0,5}{2n+1,5}$

Cioè per ricevere correttamente n bit la precisione dei due clock (che si assume uguale) deve essere $f_{nominale}(1 - k) < f_{reale} < f_{nominale}(1 - k)$, con $k < \frac{0,5}{2n+1,5}$.

Di conseguenza, guardando il problema inverso e cioè data la precisione dei due clock, ricavare il massimo numero di bit tra uno Start-bit ed uno Stop bit. Ovviamente si ha:

$$n < \frac{0,5 \quad 1,5k}{2k}$$

Analizziamo ora il caso in cui la precisione dei due clock è diversa. In particolare:

$$f_n(1 - k) < f_t < f_n(1 + k)$$
$$f_n(1 - h) < f_r < f_n(1 + h)$$

Si ha:

$$\frac{1 - k}{1 + h} < \frac{f_t}{f_r} < \frac{1 + k}{1 - h}$$

Di conseguenza per ricevere correttamente n bit si deve avere:

$$\left(\frac{f_t}{f_r}\right)^{MAX} = \frac{1 + k}{1 - h} < \frac{n+1}{n+0,5}$$
$$\left(\frac{f_t}{f_r}\right)^{MIN} = \frac{1 - k}{1 + h} > \frac{n}{n+0,5}$$

Supponiamo di conoscere *k* ed *n*:

Si ha: $1 - h > (1 + k)\frac{n+0,5}{n+1}$ e di conseguenza: $h < \frac{0,5 - k(n+0,5)}{n+1}$, e

$1 + h < (1 - k)\frac{n+0,5}{n}$ e di conseguenza: $h < \frac{0,5 - k(n+0,5)}{n}$.

La più conservativa delle due diseguaglianze è la seconda.

4. Appunto sulla Codifica Manchester

In questo paragrafo si prende in esame il problema della codifica sincrona Manchester. Data la durata di un bit, lo zero viene codificato attraverso un valore zero nella prima meta' del bit ed un valore uno nella seconda meta' del bit, e viceversa per il valore uno. Il programma di seguito proposto affronta il tema della sincronizzazione. Diversamente dal caso asincrono, qui il trasmettitore ed il ricevitore non condividono la conoscenza del clock rate.

```
   // shbt: Sync. Half Bit Time: Shared Variable
   // bp: Bit Pulse: Shared Variable

   state=S0;
   while (TRUE){
       event=wait_event();
       switch(state){
   // Inizio: attesa del primo fronte down to up
      case S0:   switch(event){
                       case dtu:  dtut=gettime();
                                  state=S1;
                 }
                 break;
//    Ricevuto down to up: attesa del fronte up to down
      successivo
      case S1:   switch(event){
                       case utd:  utdt=gettime();
                                  up0=utdt-dtut;
                                  state=S2;
                 }
                 break;
//    Ricevuti dtu+utd: attesa della prossima sequenza
      case S2:   switch(event){
                       case dtu:  dtut=gettime();
                                  state=S3;
                                  break;
                 }
                 break;
//    Ricevuto down to up: attesa del fronte up to down
      successivo
      case S3:   switch(event){
                       case utd:  utdt=gettime();
                                  up1=utdt-dtut;
                                  if (up1 ≈ 2*up0) ){
   // Set Shared Variable: Bit Pulse
                                        bp=up1;
   // Set Shared Variable. Sync. previous hbt
                                        shbt=utdt;
                                        issue_event(NEW_SYNC);
                                        up0=up1/2;
                                  } else up0=up1;
```

```
                                    state=S2;
                }
                break;
        }
    }
```

```
// shbt: Sync. Half Bit Time: Shared Variable
// bp: Bit Pulse: Shared Variable
// nsbt: Next Sampling Bit Time

while (TRUE){
    event=wait_event();
    switch(event){
              case NEW_SYNC:
                              nsbt=(shbt+bp*3/4);
                              reset_alarm(SBT);
                              set_alarm(SBT, nsbt);
                              break;

              case SBT:
                              out(get_bit));
                              nsbt= nsbt+bp;
                              set_alarm(SBT, nsbt);
                              break;
   }
}
```

5. Appunto sulla codifica AMI – HDB3

In questo paragrafo si prende in esame la codifica attraverso Alternate Mark Inversion (AMI) ed in particolare lo standard HDB3. Il problema e' quello della sostituzione di sequenze di zeri che possono dare luogo a problemi di sincronizzazione con violazioni della Alternate Mark Inversion. Il programma qui proposto codifica una sequenza di bit in una sequenza HdB3 e decodifica una sequenza HDB3 in una sequenza di bit.

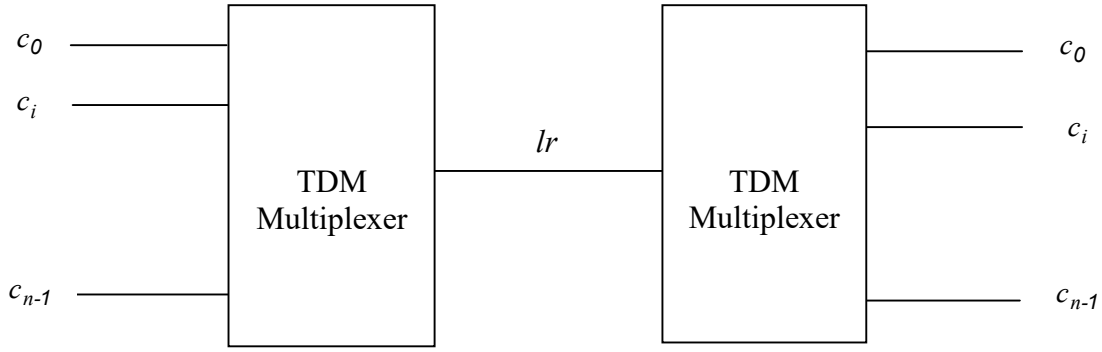
```
/*

Example 3 of HDB3 encoding

Original:     1 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 0 1 0 0 0 0

AMI:          + 0 - 0 0 0 0 + 0 0 0 0 - + 0 0 0 0 - + - 0 0 0 0 + - + - 0 0 0 0 + 0 - 0 0 0 0

HDB3:         + 0 - 0 0 0 - + 0 0 0 + - + - 0 0 - + - + 0 0 0 + - + - + - 0 0 - + 0 - + 0 0 +

*/

# define NEG 'N'
# define POS 'P'
# define ZERO '0'
# define ONE '1'


char is[];          // Input Sequence
char ts[SLEN];      // Transmitted Sequence
char rs[SLEN];      // Received Sequence
char last_pulse;
char last_violation;
```

```
code(char *is, char *ts){
     int actual_len, i=0, counter=0;
     actual_len=strlen(is);
     while (i<actual_len){
           if (is[i] == ONE) {
              counter=0;
                if (last_pulse==NEG){
                  ts[i++]=POS; last_pulse=POS;
                } else {
                  ts[i++]=NEG; last_pulse=NEG;
                }
           } else {
               ts[i++]= ZERO;
               counter++;
               if (counter == 4) {
                 if ((last_pulse==NEG) &&(last_violation==POS)) {
                    ts[i-1]=NEG; last_pulse=NEG; last_violation=NEG;
                 } else  if ((last_pulse==NEG) &&(last_violation==NEG)) {
                           ts[i-4]=POS; ts[i-1]=POS; last_pulse=POS; last_violation=POS;
                          } else if ((last_pulse==POS) && (last_violation==POS)) {
                                    ts[i-4]=NEG; ts[i-1]=NEG; last_pulse=NEG;last_violation=NEG;
                                  } else if ((last_pulse==POS) && (last_violation==NEG)) {
                                             ts[i-1]=POS; last_pulse=POS; last_violation=POS;
                                           }
                 counter=0;
              }
           }
       }
       ts[i]='\0';
}
```

```
decode(char *ts, char *rs){
     int actual_len, i=0, counter=0;
     actual_len=strlen(ts);
     while (i<actual_len){
           if ((ts[i]==POS) ||(ts[i]==NEG)) rs[i]=ONE;
           else rs[i]=ZERO;
           if (i > 3) {
              if (((ts[i-3]==POS) && (ts[i-2]==ZERO) && (ts[i-1]==ZERO) && (ts[i]==POS))||
                  ((ts[i-3]==NEG) && (ts[i-2]==ZERO) && (ts[i-1]==ZERO) && (ts[i]==NEG))||
                  ((ts[i-4]==NEG) && (ts[i-3]==ZERO) && (ts[i-2]==ZERO) && (ts[i-1]==ZERO) &&
                    (ts[i]==NEG))||
                  ((ts[i-4]==POS) && (ts[i-3]==ZERO) && (ts[i-2]==ZERO) && (ts[i-1]==ZERO) &&
                   (ts[i]==POS))) {
                        rs[i-3]=ZERO;
                        rs[i]=ZERO;
              }
           }
           i++;
     }
     rs[i]='\0';
}
```

6. Appunto su Time Division Multiplexing

In questo paragrafo si prende in esame il tema del multipexing nel tempo, con particolare riferimento alla condivisione di una linea digitale per la trasmissione di piu' linee digitali tributarie. Il problema, in estrema sintesi, e' quello della sincronizzazione a livello di frame, dove un frame e' un aggregato di bit che contiene alcuni bit di ciascuna line tributaria ed alcuni bit per la sincronizzazione (Path Overhead per Frame).



Parametri:

n : Numero di circuiti

c_i : Canale tributario *i*

r_i : Rate del circuito *i*

sl : Lunghezza della Sequenza di Sincronizzazione

pof : Path Overhead per Frame

k : costante arbitraria

Da cui si possono ricavare:

nfs : Numero di Frame necessari per la Sincronizzazione

fl : Lunghezza del singolo Frame

s_i : Size della slot relativa al circuito *i* nel frame

fr : Frame Rate

lr : Line Rate

Si ha:

1) $nfs = \frac{sl}{pof}$, tenendo presente che la sequenza di sincronizzazione è divisa tra i frame.

2) $fl = \sum_{i=0}^{n-1} s_i + pof$, tenendo presente che ogni frame include i canali tributari e il *pof*.

3) $s_i = \frac{r_i}{MCD(r_j, j = 0,\ldots,n-1)} k$, dove $k \geq 1$ è una costante arbitraria e *MCD* indica il Massimo Comun Divisore.

4) $fr = \frac{MCD(r_j, j = 0,...,n-1)}{k}$

5) $lr = fr * fl = \frac{MCD(r_j, j = 0,...,n \quad 1)}{k}(\frac{\sum_{i=0}^{n-1} r_i}{MCD(r_j, j = 0,...,n-1)}k + pof) =$

$$= (\sum_{i=0}^{n-1} r_i + \frac{1}{k} MCD(r_j, j = 0,...,n-1) pof)$$

ESEMPIO: T1

Nel caso del T1 si ha:

Parametri:

n :	Numero di circuiti: $n = 24$
r_i :	Rate del circuito $r_i = 64 Kbps, \quad i = 0, n-1$
sl :	Lunghezza della Sequenza di Sincronizzazione: $sl = 16$
pof :	Path Overhead per Frame: $pof = 1$
k :	costante arbitraria: $k = 8$

e di conseguenza:

$$nfs = \frac{16\ bit}{\frac{1\ bit}{frame}} = 16\ frame$$

$$fl = \frac{24\ bit}{frame} \bullet 8 + \frac{1\ bit}{frame} = 193\ bit$$

$$s_i = \frac{64\ Kbps}{64\ Kbps} \bullet 8$$

$$fr = \frac{64\ k}{8} = 8\ kfps$$

$$lr = 8\ kfps \bullet \frac{193\ bit}{frame} = 1544\ Kbps$$

7. Appunto su Rate di Livello 2 in presenza di Errori

In questo paragrafo si prende in esame il problema della dipendenza del rate di livello 2, che tiene conto delle ritrasmissioni e dei timeout dovuti agi errori. Intuitivamente il rate di livello 2 tende al rate di livello 1 in assenza di errori e decresce con il Bit Error Rate.

Il Rate di Livello 2 nel PAR in presenza di errori può essere rappresentato come segue:

$$Rate_{Liv\,2}(s) = \frac{(1 - s \cdot BER) \cdot s}{(1 - s \cdot BER) \cdot RTT + s \cdot BER \cdot \mathrm{TimeOut}}$$

Supponiamo: $\mathrm{TimeOut} = \alpha \cdot \mathrm{RTT}$. Ad esempio se $\alpha = 2$ il TimeOut è il doppio del RTT.

$$Rate_{Liv\,2}(s) = \frac{1}{\mathrm{RTT}} \frac{s - \mathrm{BER} \cdot s^2}{1 + \mathrm{BER} \cdot (\alpha - 1) \cdot s}$$

Troviamo il valore di s che massimizza il Rate di Livello 2.

$$\frac{d\,Rate_{Liv\,2}(s)}{d\,s} = 0$$

$$(1 - 2 \cdot BER \cdot s) \cdot (1 + BER\,(\alpha - 1) \cdot s) - \cdot BER\,(\alpha - 1)(s - \mathrm{BER} \cdot s^2) = 0$$

$$1 + BER\,(\alpha - 1) \cdot s - 2 \cdot BER \cdot s - 2 \cdot BER^2(\alpha - 1)s^2 - BER\,(\alpha - 1) \cdot s + BER^2(\alpha - 1)s^2 = 0$$

$$1 - 2 \cdot BER \cdot s - BER^2(\alpha - 1)s^2 = 0$$

$$BER^2(\alpha - 1)s^2 + 2 \cdot BER \cdot s - 1 = 0$$

$$s = \frac{-2 \cdot BER \pm \sqrt{4 \cdot BER^2 + 4 \cdot BER^2 \cdot (\alpha - 1)}}{2 \cdot BER^2(\alpha - 1)}$$

$$s = \frac{-1 \pm \sqrt{\alpha}}{BER \cdot (\alpha - 1)}$$

Poiché deve essere $s > 0$, l'unica soluzione valida è:

$$s = \frac{-1 + \sqrt{\alpha}}{BER \cdot (\alpha - 1)}$$

Ad esempio se $BER = 10^{-6}$ e $\alpha = 2$, si ha $s = 0{,}41 \cdot 10^6$, e cioè il valore di s che minimizza il rate di livello 2 è 410Kb e cioè circa 50KBytes.

8. Esempio di codice a supporto di Inter Process Communication

In questo paragrafo si prende in esame il tema della comunicazione tra processi, all'interno di un sistema operativo, a supporto dell'interazione tra entita' di diversi livelli. In particolare si introduce il concetto di "coda di ingresso" e si elencano e descrivono gli eventi associati alla gestione di tale coda.

In questa implementazione esiste un'unica lista di processi "waiting su target process input full",chiamata wtpiqf list. Quando un processo A scrive ad un processo B che ha la input queue full, il processo A viene messo in stato WTPIQF ed un item <source process_pid:A, target_process pid: B> viene aggiunto alla wtpiqf list.

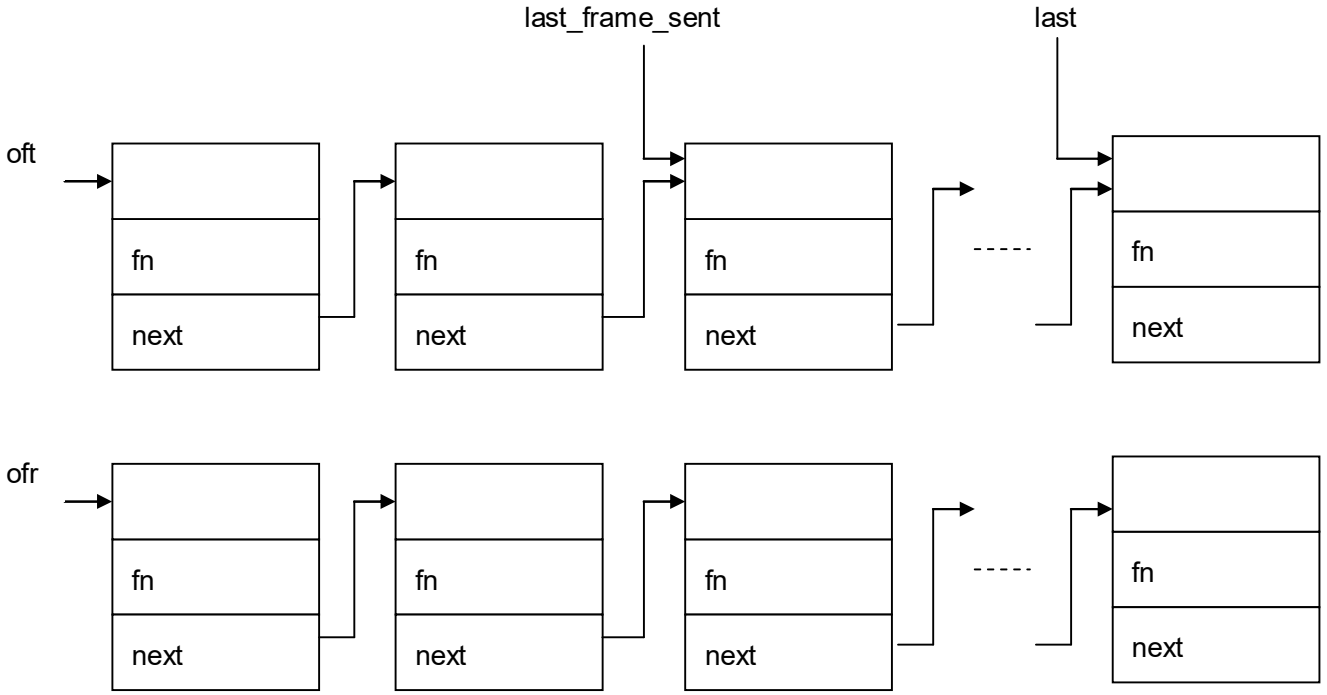
```
struct {
   pid source_process, target_process;
   …
}


write_target_process_input_queue(target pid) {
        if (target_process input_queue == AVAILABLE) {
              enqueue in target_process input_queue from process space buffer;
              if (target_process status = WIQE) {
                    dequeue to target_process space buffer;
                    set target_process status = READY;
              }
              return;
        }
        if (target_process input_queue == FULL) {
              set calling_process status = WTPIQF;
              add <calling_process pid, target_process pid > to wtpiqf pid list
              }
}
```

Quando un calling_process richiede una lettura dalla coda di ingresso, se la coda è vuota il calling_process viene messo in uno stato chiamato WIQE (Wait Input Queue Empty). Altrimenti, se la coda non è piena viene effettuato il dequeue del messaggio mentre se la coda è piena viene fatto il dequeue del messaggio e la ricerca di un eventuale elemento nella coda wtpiqf avente il calling_process come target_process. Se l'elemento esiste, viene fatto l'enqueue del messaggio dal source_process space buffer nell'input_queue del calling_process e il source_process viene messo nello stato READY.

```
read_from_process_input_queue(){
        if (calling_process input queue EMPTY){
             set calling_process status = WIQE;
             return;
        }
        if (calling process input queue not EMPTY){
             dequeue to process space buffer;
             if (calling process input queue FULL){
                  for (scan wtpiqf process list) {
                       if (list->target process == calling_process) {
                            enqueue    from list->source_process space buffer;
                                       in calling_process input_queue
                            set list->source_process status = READY;
                            remove entry from wtpiqf pid list;
                       }
                  }
                  return;
             }
}
```

9. Esempio di Data Link Layer Entity con Window Flow Control

In questo paragrafo viene evidenziata l'architettura di una Data Link Layer Entity in presenza di controllo di flusso. Il tema e' quello delle finestre di trasmissione e di ricezione, e quindi della gestione degi acknowledge per aggiornare lo stato della trasmissione e della ricezione.



```
// Qui si assume che l'IPC preveda per ogni processo una coda diversa
// per ogni possibile processo inviante

// oft: pointer to Oldest Frame in Tx-Win
// ofr: pointer to Oldest Frame in Rx-Win
// next_seq_num: Next sequence number to be assigned
// last_frame_sent: pointer to last frame sent to PL
// last: pointer to last frame received from NL
// frame_expected: sequence number of next frame expected

void from_NL () {
/* Receive from IPC */
   f=malloc(SIZE);
   receive_from(NL, f->payload);
/* Build the frame */
   f->fn=next_seq_num; f->an=frame_expected; f->next=NIL;

   next_seq_num=next_seq_num++;                    /* Circular increment */

/* Insert Frame in Tx_Win */
   if (oft == NULL) {                          /* Tx-Win Empty*/
        oft=f;
        last=f;
   } else {                                         /* Tx-Win Not Empty*/
        last->next=f;
        last=f;
   }
   if ((f->fn - oft->fn)< WIN_SIZE) {          /* Circular comparison */
        set_timer(get_time()+TIMEOUT, f);
        send_to (PL, f);
        last_frame_sent =f;
}
}
```

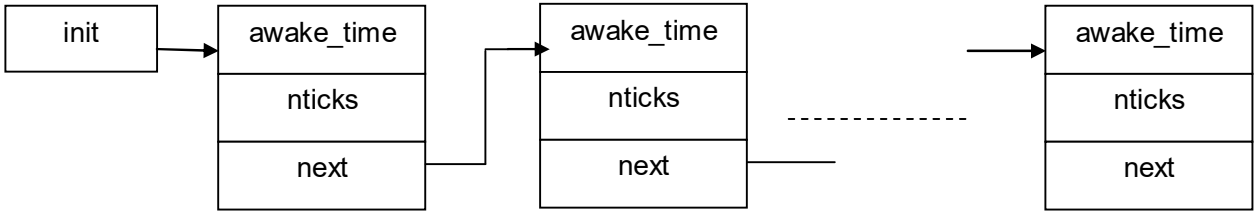
```
void timer(frame *f){
set_timer(get_time()+TIMEOUT, f);
send_to (PL, f);
}
```

```
void from_PL (){
/* Receive from IPC */
f=malloc(SIZE);
receive_from(PL, f);
f->next=NIL;

/* ACK Processing - ift: Oldest Frame in Tx_win  */
while (oft && (oft->fn < f->an)) {         /* Circular comparison */ /* Flush Tx Win */
        tmp=oft;
        oft=oft->next;
        free(tmp);
           if (last_frame_sent->next) {
              set_timer(get_time()+TIMEOUT, last_frame_sent->next);
              send_to (PL, f);
              last_frame_sent=last_frame_sent->next;
        }
}

/* Insert Frame in Rx_Win */
if (f->fn < frame_expected) {            /* Drop frame if duplicate */
} else {
        if (ofr == NIL) {                /* Rx-Win Empty */
              ofr = f;
        } else {                                 /* Rx-Win Not Empty - Sorting by insertion */
              if (f->fn < ofr->fn) {             /* First in queue */
                    f->next=ofr;                 /* Insert frame f as first */
                    ofr=f;
              } else {
                    p=ofr;                     /* Insert frame f after first */
                    while (p && (p->fn < f->fn)) {
                          q=p;
                          p=p->next;
                    }
```

```
                    q->next=f;
                    f->next=p;
                }
        }
/* Deliver frames to Network Layer */
        while (ofr && (ofr->fn == frame_expected){  /* Flush RX Win */
              send_to(nle_pid(ofr->ulp, ofr->payload);
          frame_expected = frame_expected++;        /* Circular increment */
          tmp=ofr;
          ofr=ofr->next;
          free(tmp);
        }
   }
}
```

10. Appunto su Calendar Queue Algorithm

In questo paragrafo si descrive la struttura dati Calendar Queue e le primitive di controllo della stessa. L'obiettivo e' quello di aggiungere eventi alla lista di eventi da processare e di consumare gli eventi in testa alla lista al momento opportuno.




```
// Procedure to set a timer in a Calendar Queue
set_timer (awake_time){
   if (n_elements==0) {                              // Calendar Queue empty
        init=create_cqi();
        init->awake_time=awake_time;
        init->nticks= (awake_time-current_time())/TICK_LEN;
        return;
   }
   if (n_elements > 0) {                             // Calendar Queue not empty
        if (awake_time <= init->awake_time) {       // New element first
             t=create_cqi();
             t->awake_time = awake_time;
             t->nticks = (awake_time-current_time())/TICK_LEN;
             t->next=init;
             init=t;
             init->next->nticks=
                  (init->next->awake_time-init->awake_time)/TICK_LEN;
```

```
        }

              if (awake_time > init->awake_time) {        // New element not first
              p=init;
              while (p && (p->awake_time < awake_time)){
                    q=p;
                    p=p->next;
              }
// Either p = NIL or q->awake_time < awake_time < p->awake_time
// In any case new element after q
              t=create_cqi();
              t->awake_time = awake_time;
              t->nticks = (awake_time-q->awake_time)/TICK_LEN;
              q->next=t;
              t->next=p;
              if (p) {
                    p->nticks=(p->awake_time-awake_time)/TICK_LEN;
              }
        }
   }
   ++n_elements;
}

cq_tick(){
   if (n_elements ==0) return;
   p=init;
   p->nticks--;
   while ((p->nticks ==0) && p){
        consume(p);
        p=p->next;
        --n_elements;
   }
}
```

11. Appunto sulla Error Correction Code

In questo paragrafo si presenta un esempio di ridondanza nella trasmissione in modo tale da consentire la correzione dell'errore da parte del ricevitore.

Supponiamo di avere i seguente codice:

Messaggio	Codeword
00	00000 00000
01	00000 11111
10	11111 00000
11	11111 00000

Supponiamo che venga trasmessa la sequenza : 00000 11111. Se viene ricevuta senza errori si ha:

Ricezione di 00000 11111	Distanza dalle codeword
Distanza da 00000 00000	5
Distanza da 00000 11111	0
Distanza da 11111 00000	10
Distanza da 11111 11111	5

Ricezione di 00001 11111	Distanza dalle codeword
Distanza da 00000 00000	6
Distanza da 00000 11111	1
Distanza da 11111 00000	9
Distanza da 11111 11111	4

Ricezione di 00001 01111	Distanza dalle codeword
Distanza da 00000 00000	5
Distanza da 00000 11111	2
Distanza da 11111 00000	8
Distanza da 11111 11111	5

Ricezione di 01001 01111	Distanza dalle codeword
Distanza da 00000 00000	6
Distanza da 00000 11111	3
Distanza da 11111 00000	7
Distanza da 11111 11111	4

Ricezione di 01001 01101	Distanza dalle codeword
Distanza da 00000 00000	5
Distanza da 00000 11111	4
Distanza da 11111 00000	6
Distanza da 11111 11111	5

Ricezione di 00000 00011	Distanza dalle codeword
Distanza da 00000 00000	2

Distanza da 00000 11111	3
Distanza da 11111 00000	7
Distanza da 11111 11111	8

In sintesi se la HD minima tra due configurazioni è HD_{MIN} (supponendo che HD_{MIN} sia dispari) allora il massimo numero di errori E_{MAX} accettabile è pari a $E_{NAX} \leq \frac{HD_{MIN} - 1}{2}$. In fatti un numero di errori $E_{NAX} > \frac{HD_{MIN} - 1}{2}$ potrebbe portare la configurazione ricevuta ad essere più vicina ad una codeword diversa da quella trasmessa che alla codeword trasmessa. La tabella sottostante mostra questo caso.

Trasmissione di 00000 11111	
Ricezione di 00000 00011	Distanza dalle codeword
Distanza da 00000 00000	2
Distanza da 00000 11111	3
Distanza da 11111 00000	7
Distanza da 11111 11111	8

12. Appunto sulla performance di CSMA/CD

In questo paragrafo si presenta una trattazione quantitativa ella performance di una rete CSMA/CD.

Stimiamo innanzitutto la probabilità di successo di una trasmissione su mezzo condiviso, che chiamiamo $\mathbf{A(k,p)}$, dove *p* è la probabilità di trasmissione della singola workstation e *k* è il numero di workstation collegate al mezzo condiviso.

Si ha:

$$\mathbf{A(k,p) = k \cdot p \cdot (1-p)^{k-1}}$$

Il fatto che $\lim_{p\to 0} A(k,p) = 0$ indica che se le workstation trasmettono con probabilità bassa il successo della trasmissione sul mezzo condiviso è basso perché non c'è alcuna trasmissione. Viceversa il fatto che $\lim_{p\to 1} A(k,p) = 0$ indica che se le workstation trasmettono con probabilità elevata il successo della trasmissione sul mezzo condiviso è basso per la presenza di collisioni.

Fissato il numero di workstation cerchiamo la probabilità di trasmissione *p* della singola workstation che massimizza la probabilità di successo della trasmissione sul mezzo condiviso.

Poniamo quindi:

$$\frac{\partial A}{\partial p} = 0$$

E quindi:

$$\frac{\partial A}{\partial p} = k((1-p)^{k-1} - p(k-1)(1-p)^{k-2}) = k(1-p)^{k-2}(1-p-pk+p)$$

$$\frac{\partial A}{\partial p} = 0 \overset{yields}{\rightarrow} p = \frac{1}{k}$$

Calcoliamo ora il valore massimo di A e cioè i valore di A nel caso in cui $p = \frac{1}{k}$.

$$A\left(k,\frac{1}{k}\right) = \left(1-\frac{1}{k}\right)^{k-1}$$

$$\lim_{k\to\infty} A\left(k,\frac{1}{k}\right) = \lim_{k\to\infty}\left(1-\frac{1}{k}\right)^{k-1} = \lim_{k\to\infty}\frac{\left(1-\frac{1}{k}\right)^{k}}{1-\frac{1}{k}}$$

$$= \lim_{k\to\infty}\left(1-\frac{1}{k}\right)^{k} = \lim_{k\to\infty} e^{k\ln\left(1-\frac{1}{k}\right)} = e^{\lim_{k\to\infty} k\cdot\ln\left(1-\frac{1}{k}\right)}$$

$$\lim_{k\to\infty} k\cdot\ln\left(1-\frac{1}{k}\right) = \lim_{k\to\infty}\frac{\ln\left(1-\frac{1}{k}\right)}{\frac{1}{k}} =$$

Derivando il numeratore ed il denominatore:

$$\lim_{k\to\infty}\frac{1}{\left(1-\frac{1}{k}\right)}\cdot\frac{1}{k^2}\cdot\frac{1}{\left(-\frac{1}{k^2}\right)} = \lim_{k\to\infty}\frac{-k}{k-1} = -1$$

Da cui si ha:

$$\lim_{k\to\infty} A\left(k,\frac{1}{k}\right) = e^{-1}$$

Calcoliamo ora il numero medio di ritrasmissioni a partire dalla probabilità di successo A della trasmissione sul mezzo condiviso. La probabilità $p(j)$ di *j* ritrasmissioni può essere calcolata come segue:

$$p(j) = A \cdot (1-A)^{j-1}$$

Da cui:

$$E(j) = \sum_{j=0}^{\infty} j \cdot A \cdot (1-A)^{j-1} = A \sum_{j=0}^{\infty} j \cdot (1-A)^{j-1} = (1-B) \sum_{j=0}^{\infty} j \cdot B^{j-1}$$

dove B indica la probabilità di insuccesso.

Noi sappiamo che, dato un generico *K* , la serie $\sum_{j=0}^{\infty} K^j$ può essere calcolata come segue:

$$\sum_{j=0}^{\infty} K^i \quad K \sum_{j=0}^{\infty} K^i = 1 \mid K \mid K^2 \mid K^3 \mid \cdots \quad K \quad K^2 \quad K^3 \quad \cdots = 1$$

Da cui

$$\sum_{j=0}^{\infty} K^j = \frac{1}{1-K}$$

Da cui ancora, considerando che

$$\sum_{j=0}^{\infty} j \cdot B^{j-1} = -\frac{\partial \sum_{j=0}^{\infty} B^j}{\partial j}$$

si ha:

$$(1-B) \sum_{j=0}^{\infty} j \cdot B^{j-1} = (1-B) \frac{1}{(1-B)^2} = \frac{1}{1-B}$$

Quindi si ha:

$$E(j) = \sum_{j=0}^{\infty} j \cdot A \cdot (1-A)^{j-1} = \frac{1}{A}$$

Se $A = \frac{1}{e}$, si ha $E(j) = e$.

Se poi ogni ritrasmissione richiede mediamente τ, il tempo medio di attesa è pari a $e \cdot \tau$.

13. Esempio di Bridge Code

In questo paragrafo si descrive il funzionamento di un bridge attraverso un esempio di psudocodice che ne governa le operazioni.

```
-/-> Received Frame (src_port)
{
//   Forwarding
     frame=dequeue_frame(src_port);
     if (frame.dmac == BROADCAST){
               for (i=0 ; i < total_number_of_ports ; i++)
                         if (i != src_port) enqueue (frame, i) ;
     }
     if (frame.dmac != BROADCAST){
               table_index= lookup (mac_to_port, frame.dmac);
               if (table_index != NOT_FOUND)
                         enqueue (frame, mac_to_port [table_index].port) ;
               else
                         for (i=0 ; I < total_number_of_ports ; i++)
                                   if (i != src_port)  enqueue (frame, i) ;
     }
//   Learning
     table_index= lookup(mac_to_port, frame.smac);
     if (table_index == NOT_FOUND)
               table_index= add_entry (mac_to_port);
     mac_to_port [table_index].mac=frame.smac;  mac_to_port [table_index].port =src_port;
     mac_to_port [table_index].age=0;
}
-/-> Timer_tick
{
```

```
        for (i=0; i< mac_to_port_table_len; i++)
                if (++mac_to_port [i].age >= MAX_AGE) remove_entry (mac_to_port, i);
}
```

14. Un'implementazione dell'algoritmo Shortest Path di Dijkstra

In questo paragrafo si propone un programma in C language che implementa l'algoritmo di Dijkstra per il percorso minimo.

```
int am [XXX][XXX];       // Adjacency Matrix
int pm [XXX][XXX];       // Port Matrix
int target=0;

struct {
   int cost;
   int next;
   int def;
} n[XXX];

// Gaph Labeling: from am[][] + target to n[]
//
void label() {
int i, j, min, indmin, iter;
// Initialization
   for (i=0; i < XXX ; i++) {
        n[i].cost= INF;
        n[i].next= UNK;
        n[i].def= FALSE;
   }
   n[target].cost=0;
   n[target].next=target;

   iter=0;
   do {
        n[target].def=TRUE;
        for (i=0; i < XXX; i++) {
```

```
            if (am[target][i] != INF) {
                  if ((n[target].cost+am[target][i]) < n[i].cost) {
                        n[i].cost=n[target].cost+am[target][i];
                        n[i].next=target;
                  }
            }
      }

      min=INF;
      indmin=-1;
      for (i=0; i < XXX; i++) {
            if (n[i].def == FALSE ) {
                  if (n[i].cost < min) {
                        min= n[i].cost;
                        indmin=i;
                  }
            }
      }

      target=indmin;
      iter++;
   } while (indmin !=-1);

// From labeled graph to Routing Table
for (k=0; k < XXX; k++) {
  while (n[i].next) != myself) {
    cost+=n[i]->cost;
    i=n[i].next;
  }
  cost+=n[i]->cost;
  update_Routing_Table (k, pm[k][i]);

}
```

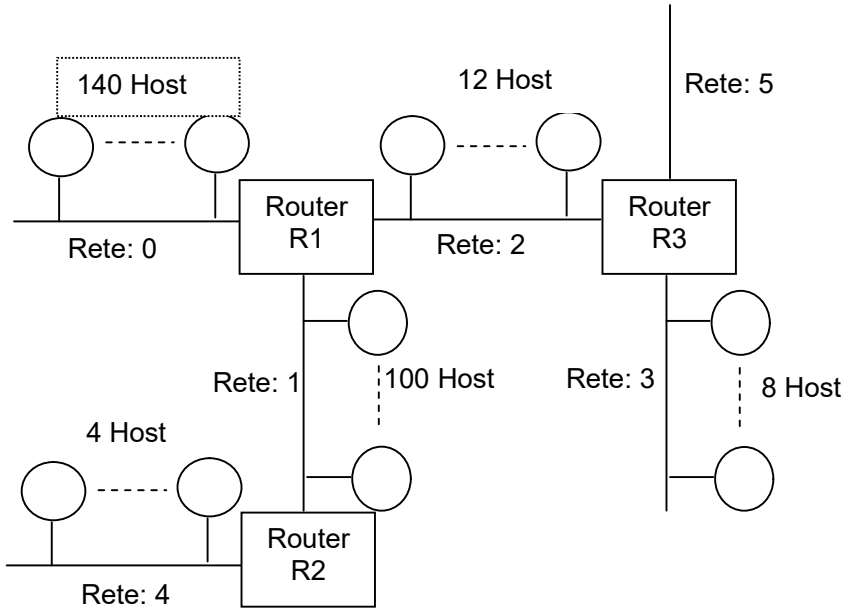
15. Esempio di algoritmo per indirizzamento IP - CIDR

In questo paragrafo si presenta un programma che consente la definizione degli indirizzi di un sistema di reti a partire dalla descrizione del sistema stesso in termini di dimensioni delle singole reti che lo compongono.

```
…
   ip[0].initial=set_bit(0, ip[0].initial);
   ip[0].final= reset_bit(0, ip[0].final);
   for (j=1; j < (32-ip[0].prefix); j++){
              ip[0].initial=reset_bit(j, ip[0].initial); ip[0].final=set_bit(j, ip[0].final);
   }
   for (j=(32-ip[0].prefix); j< 32; j++){
              ip[0].initial=reset_bit(j, ip[0].initial);
              ip[0].final=reset_bit(j, ip[0].final);
   }
   for (i=1; i<n_items; i++){
        t=ip[i-1].final;
              t=set_bit(0, t);
              for (j=1; j < (32-ip[i].prefix); j++)
                   t=reset_bit(j, t);
              for (j=32-ip[i].prefix; ((bit (j, t) == 1) && (j < 32)); j++)
                   t=reset_bit(j, t);
        t=set_bit(j, t);
        ip[i].initial=t;
        t=reset_bit(0, t);
        for (j=1; j < (32-ip[i].prefix); j++)
                   t=set_bit(j, t);
        ip[i].final=t;
   }
   for (j=31; ((bit (j, ip[n_items-1].final) != 1) && (j > 0)); j--) ;

   general_prefix=32-(j+1);
```

16. Esempio di Indirizzamento IP - CIDR

In questo paragrafo si presenta un esempio di sistema di reti e la numerazione che puo' essere applicata a tale sistema.



```
 0: n_elements: 140, prefix 24
 1: n_elements: 100, prefix 25
 2: n_elements: 12, prefix 28
 3: n_elements: 8, prefix 28
 4: n_elements: 4, prefix 29
 5: n_elements: 2, prefix 30
 General Prefix: 23

ip[0].initial: 00000000000000000000000000000001/240.0.0.1/24
ip[0].final:   00000000000000000000000011111110/240.0.0.254/24

ip[1].initial: 00000000000000000000000100000001/250.0.1.1/25
ip[1].final:   00000000000000000000000101111110/250.0.1.126/25

ip[2].initial: 00000000000000000000000110000001/280.0.1.129/28
ip[2].final:   00000000000000000000000110001110/280.0.1.142/28

ip[3].initial: 00000000000000000000000110010001/280.0.1.145/28
ip[3].final:   00000000000000000000000110011110/280.0.1.158/28

ip[4].initial: 00000000000000000000000110100001/290.0.1.161/29
ip[4].final:   00000000000000000000000110100110/290.0.1.166/29

ip[5].initial: 00000000000000000000000110101001/300.0.1.169/30
ip[5].final:   00000000000000000000000110101010/300.0.1.170/30
```

17. Esempio di acquisizione dati durante un accesso Web

Questo paragrafo contiene la traccia di una connessione TCP ottenuta attraverso Wireshark.

N.	Time	Source	Destination	Protocol	Info
1	0.000000	0.0.0.0	255.255.255.255	DHCP	DHCP Request - Transaction ID 0xee9c5955
2	0.026411	130.251.12.33	130.251.12.63	DHCP	DHCP ACK - Transaction ID 0xee9c5955
3	1.946353	Fujitsu_6e:da:aa	Broadcast	ARP	Who has 130.251.12.32? Tell 130.251.12.63
4	1.947153	Supermic_8d:ea:ba	Fujitsu_6e:da:aa	ARP	130.251.12.32 is at 00:30:48:8d:ea:ba
5	1.947159	130.251.12.63	130.251.12.32	DNS	Standard query A www.google.com
6	1.992448	130.251.12.32	130.251.12.63	DNS	Standard query response CNAME www.l.google.com
7	3.013179	Fujitsu_6e:da:aa	Broadcast	ARP	Who has 130.251.12.126? Tell 130.251.12.63
8	3.013956	Cisco_bd:6e:d8	Fujitsu_6e:da:aa	ARP	130.251.12.126 is at 00:1b:d4:bd:6e:d8
9	6.180282	130.251.12.63	209.85.129.104	TCP	[SYN] Seq=0 Win=65535 Len=0 MSS=1460
10	6.218166	209.85.129.104	130.251.12.63	TCP	[SYN, ACK] Seq=0 Ack=1 Win=5720 Len=0 MSS=1430
11	6.218242	130.251.12.63	209.85.129.104	TCP	[ACK] Seq=1 Ack=1 Win=65535 Len=0
12	6.218380	130.251.12.63	209.85.129.104	HTTP	GET / HTTP/1.1
13	6.256571	209.85.129.104	130.251.12.63	TCP	[ACK] Seq=1 Ack=748 Win=6723 Len=0
14	6.264288	209.85.129.104	130.251.12.63	HTTP	HTTP/1.1 302 Found (text/html)
15	6.267622	130.251.12.63	130.251.12.32	DNS	Standard query A www.google.it
16	6.268887	130.251.12.32	130.251.12.63	DNS	Standard query response CNAME www.google.com
17	6.269719	130.251.12.63	209.85.129.147	TCP	[SYN] Seq=0 Win=65535 Len=0 MSS=1460
18	6.307816	209.85.129.147	130.251.12.63	TCP	[SYN, ACK] Seq=0 Ack=1 Win=5720 Len=0 MSS=1430
19	6.307863	130.251.12.63	209.85.129.147	TCP	[ACK] Seq=1 Ack=1 Win=65535 Len=0
20	6.307960	130.251.12.63	209.85.129.147	HTTP	GET / HTTP/1.1
21	6.347174	209.85.129.147	130.251.12.63	TCP	[ACK] Seq=1 Ack=878 Win=7016 Len=0
22	6.402030	209.85.129.147	130.251.12.63	TCP	[TCP segment of a reassembled PDU]
23	6.402054	209.85.129.147	130.251.12.63	TCP	[TCP segment of a reassembled PDU]
24	6.402065	209.85.129.147	130.251.12.63	HTTP	HTTP/1.1 200 OK (text/html)
25	6.402090	130.251.12.63	209.85.129.147	TCP	[ACK] Seq=878 Ack=3068 Win=65535 Len=0
26	6.436849	130.251.12.63	209.85.129.147	HTTP	GET /intl/it_it/images/logo.gif HTTP/1.1
27	6.475293	209.85.129.147	130.251.12.63	TCP	[ACK] Seq=3068 Ack=1783 Win=9050 Len=0
28	6.479127	209.85.129.147	130.251.12.63	HTTP	HTTP/1.1 304 Not Modified
29	6.554985	130.251.12.63	209.85.129.147	HTTP	GET /images/nav_logo3.png HTTP/1.1
30	6.598081	209.85.129.147	130.251.12.63	HTTP	HTTP/1.1 304 Not Modified
31	6.747719	130.251.12.63	209.85.129.147	TCP	[ACK] Seq=2682 Ack=3318 Win=65285 Len=0
32	11.269547	Supermic_8d:ea:ba	Fujitsu_6e:da:aa	ARP	Who has 130.251.12.63? Tell 130.251.12.33
33	11.269564	Fujitsu_6e:da:aa	Supermic_8d:ea:ba	ARP	130.251.12.63 is at 00:17:42:6e:da:aa
34	12.107134	130.251.12.63	147.162.2.90	TCP	[SYN] Seq=0 Win=65535 Len=0 MSS=1460

```
Frame 1
0000  ff ff ff ff ff ff 00 17 42 6e da aa 08 00 45 00   ........Bn....E.
0010  01 5c 67 ec 00 00 80 11 d1 a5 00 00 00 00 ff ff   .\g.............
0020  ff ff 00 44 00 43 01 48 0c 24 01 01 06 00 ee 9c   ...D.C.H.$......
0030  59 55 05 00 00 00 00 00 00 00 00 00 00 00 00 00   YU..............
0040  00 00 00 00 00 00 00 17 42 6e da aa 00 00 00 00   ........Bn......
0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00a0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00b0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00c0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00e0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00f0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0110  00 00 00 00 00 00 63 82 53 63 35 01 03 3d 07 01   ......c.Sc5..=..
0120  00 17 42 6e da aa 32 04 82 fb 0c 3f 0c 0f 46 53   ..Bn..2....?..FS
0130  43 30 38 31 32 31 34 30 37 32 34 31 31 51 13 00   C081214072411Q..
0140  00 00 46 53 43 30 38 31 32 31 34 30 37 32 34 31   ..FSC08121407241
0150  31 2e 3c 08 4d 53 46 54 20 35 2e 30 37 0b 01 0f   1.<.MSFT 5.07...
0160  03 06 2c 2e 2f 1f 21 f9 2b ff                     ..,./.!.+.

Frame 2
0000  00 17 42 6e da aa 00 30 48 8d ea ba 08 00 45 10   ..Bn...0H.....E.
0010  01 68 00 00 00 00 10 11 8b 1f 82 fb 0c 21 82 fb   .h...........!..
0020  0c 3f 00 43 00 44 01 54 df 25 02 01 06 00 ee 9c   .?.C.D.T.%......
0030  59 55 05 00 00 00 00 00 00 00 82 fb 0c 3f 00 00   YU...........?..
0040  00 00 00 00 00 00 00 17 42 6e da aa 00 00 00 00   ........Bn......
0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00a0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00b0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00c0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00e0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00f0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0110  00 00 00 00 00 00 63 82 53 63 35 01 05 36 04 82   ......c.Sc5..6..
0120  fb 0c 21 33 04 00 00 54 60 51 20 03 02 02 46 53   ..!3...T`Q ...FS
0130  43 30 38 31 32 31 34 30 37 32 34 31 31 2e 63 69   C081214072411.ci
0140  70 69 2e 75 6e 69 67 65 2e 69 74 01 04 ff ff ff   pi.unige.it.....
0150  80 0f 0d 63 69 70 69 2e 75 6e 69 67 65 2e 69 74   ...cipi.unige.it
0160  03 04 82 fb 0c 7e 06 04 82 fb 0c 20 2c 04 82 fb   .....~..... ,...
0170  0c 0c 2e 01 02 ff                                 ......

Frame 3
0000  ff ff ff ff ff ff 00 17 42 6e da aa 08 06 00 01   ........Bn......
0010  08 00 06 04 00 01 00 17 42 6e da aa 82 fb 0c 3f   ........Bn.....?
0020  00 00 00 00 00 00 82 fb 0c 20                     .........
```

```
Frame 4
0000  00 17 42 6e da aa 00 30 48 8d ea ba 08 06 00 01   ..Bn...0H.......
0010  08 00 06 04 00 02 00 30 48 8d ea ba 82 fb 0c 20   .......0H...... 
0020  00 17 42 6e da aa 82 fb 0c 3f 00 00 00 00 00 00   ..Bn.....?......
0030  00 00 00 00 00 00 00 00 00 00 00 00               ............


Frame 5
0000  00 30 48 8d ea ba 00 17 42 6e da aa 08 00 45 00   .0H.....Bn....E.
0010  00 3c 68 09 00 00 80 11 b4 52 82 fb 0c 3f 82 fb   .<h......R...?..
0020  0c 20 04 01 00 35 00 28 2e 66 20 05 01 00 00 01   . ...5.(.f .....
0030  00 00 00 00 00 00 03 77 77 77 06 67 6f 6f 67 6c   .......www.googl
0040  65 03 63 6f 6d 00 00 01 00 01                     e.com.....


Frame 6
0000  00 17 42 6e da aa 00 30 48 8d ea ba 08 00 45 00   ..Bn...0H.....E.
0010  00 f0 00 00 40 00 40 11 1b a8 82 fb 0c 20 82 fb   ....@.@...... ..
0020  0c 3f 00 35 04 01 00 dc 34 e8 20 05 81 80 00 01   .?.5....4. .....
0030  00 04 00 07 00 00 03 77 77 77 06 67 6f 6f 67 6c   .......www.googl
0040  65 03 63 6f 6d 00 00 01 00 01 c0 0c 00 05 00 01   e.com...........
0050  00 02 41 ec 00 08 03 77 77 77 01 6c c0 10 c0 2c   ..A....www.l...,
0060  00 01 00 01 00 00 01 2c 00 04 d1 55 81 68 c0 2c   .......,...U.h.,
0070  00 01 00 01 00 00 01 2c 00 04 d1 55 81 63 c0 2c   .......,...U.c.,
0080  00 01 00 01 00 00 01 2c 00 04 d1 55 81 93 c0 30   .......,...U...0
0090  00 02 00 01 00 00 eb 6d 00 04 01 67 c0 30 c0 30   .......m...g.0.0
00a0  00 02 00 01 00 00 eb 6d 00 04 01 65 c0 30 c0 30   .......m...e.0.0
00b0  00 02 00 01 00 00 eb 6d 00 04 01 63 c0 30 c0 30   .......m...c.0.0
00c0  00 02 00 01 00 00 eb 6d 00 04 01 64 c0 30 c0 30   .......m...d.0.0
00d0  00 02 00 01 00 00 eb 6d 00 04 01 61 c0 30 c0 30   .......m...a.0.0
00e0  00 02 00 01 00 00 eb 6d 00 04 01 66 c0 30 c0 30   .......m...f.0.0
00f0  00 02 00 01 00 00 eb 6d 00 04 01 62 c0 30         .......m...b.0

Frame 7
0000  ff ff ff ff ff ff 00 17 42 6e da aa 08 06 00 01   ........Bn......
0010  08 00 06 04 00 01 00 17 42 6e da aa 82 fb 0c 3f   ........Bn.....?
0020  00 00 00 00 00 00 82 fb 0c 7e                     .........~


Frame 8
0000  00 17 42 6e da aa 00 1b d4 bd 6e d8 08 06 00 01   ..Bn......n.....
0010  08 00 06 04 00 02 00 1b d4 bd 6e d8 82 fb 0c 7e   ..........n....~
0020  00 17 42 6e da aa 82 fb 0c 3f 00 00 00 00 00 00   ..Bn.....?......
0030  00 00 00 00 00 00 00 00 00 00 00 00               ............


Frame 9
0000  00 1b d4 bd 6e d8 00 17 42 6e da aa 08 00 45 00   ....n...Bn....E.
0010  00 30 68 7b 40 00 80 06 b0 54 82 fb 0c 3f d1 55   .0h{@....T...?.U
0020  81 68 09 1d 00 50 f3 e5 92 0b 00 00 00 00 70 02   .h...P........p.
0030  ff ff 11 c9 00 00 02 04 05 b4 01 01 04 02         ..............


Frame 10
0000  00 17 42 6e da aa 00 1b d4 bd 6e d8 08 00 45 00   ..Bn......n...E.
0010  00 30 4e ec 00 00 30 06 59 e4 d1 55 81 68 82 fb   .0N...0.Y..U.h..
```

```
0020  0c 3f 00 50 09 1d b1 ab 66 66 f3 e5 92 0c 70 12   .?.P....ff....p.
0030  16 58 e3 6b 00 00 02 04 05 96 01 01 04 02        .X.k..........


Frame 11
0000  00 1b d4 bd 6e d8 00 17 42 6e da aa 08 00 45 00   ....n...Bn....E.
0010  00 28 68 7c 40 00 80 06 b0 5b 82 fb 0c 3f d1 55   .(h|@....[...?.U
0020  81 68 09 1d 00 50 f3 e5 92 0c b1 ab 66 67 50 10   .h...P......fgP.
0030  ff ff e2 12 00 00                                 ......


Frame 12
0000  00 1b d4 bd 6e d8 00 17 42 6e da aa 08 00 45 00   ....n...Bn....E.
0010  03 13 68 7d 40 00 80 06 ad 6f 82 fb 0c 3f d1 55   ..h}@....o...?.U
0020  81 68 09 1d 00 50 f3 e5 92 0c b1 ab 66 67 50 18   .h...P......fgP.
0030  ff ff e4 fd 00 00 47 45 54 20 2f 20 48 54 54 50   ......GET / HTTP
0040  2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 77 77 77 2e   /1.1..Host: www.
0050  67 6f 6f 67 6c 65 2e 63 6f 6d 0d 0a 55 73 65 72   google.com..User
0060  2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f   -Agent: Mozilla/
0070  35 2e 30 20 28 57 69 6e 64 6f 77 73 3b 20 55 3b   5.0 (Windows; U;
0080  20 57 69 6e 64 6f 77 73 20 4e 54 20 35 2e 31 3b    Windows NT 5.1;
0090  20 69 74 3b 20 72 76 3a 31 2e 38 2e 31 2e 31 34    it; rv:1.8.1.14
00a0  29 20 47 65 63 6b 6f 2f 32 30 30 38 30 34 30 34   ) Gecko/20080404
00b0  20 46 69 72 65 66 6f 78 2f 32 2e 30 2e 30 2e 31    Firefox/2.0.0.1
00c0  34 0d 0a 41 63 63 65 70 74 3a 20 74 65 78 74 2f   4..Accept: text/
00d0  78 6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f   xml,application/
00e0  78 6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f   xml,application/
00f0  78 68 74 6d 6c 2b 78 6d 6c 2c 74 65 78 74 2f 68   xhtml+xml,text/h
0100  74 6d 6c 3b 71 3d 30 2e 39 2c 74 65 78 74 2f 70   tml;q=0.9,text/p
0110  6c 61 69 6e 3b 71 3d 30 2e 38 2c 69 6d 61 67 65   lain;q=0.8,image
0120  2f 70 6e 67 2c 2a 2f 2a 3b 71 3d 30 2e 35 0d 0a   /png,*/*;q=0.5..
0130  41 63 63 65 70 74 2d 4c 61 6e 67 75 61 67 65 3a   Accept-Language:
0140  20 69 74 2d 69 74 2c 69 74 3b 71 3d 30 2e 38 2c    it-it,it;q=0.8,
0150  65 6e 2d 75 73 3b 71 3d 30 2e 35 2c 65 6e 3b 71   en-us;q=0.5,en;q
0160  3d 30 2e 33 0d 0a 41 63 63 65 70 74 2d 45 6e 63   =0.3..Accept-Enc
0170  6f 64 69 6e 67 3a 20 67 7a 69 70 2c 64 65 66 6c   oding: gzip,defl
0180  61 74 65 0d 0a 41 63 63 65 70 74 2d 43 68 61 72   ate..Accept-Char
0190  73 65 74 3a 20 49 53 4f 2d 38 38 35 39 2d 31 2c   set: ISO-8859-1,
01a0  75 74 66 2d 38 3b 71 3d 30 2e 37 2c 2a 3b 71 3d   utf-8;q=0.7,*;q=
01b0  30 2e 37 0d 0a 4b 65 65 70 2d 41 6c 69 76 65 3a   0.7..Keep-Alive:
01c0  20 33 30 30 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e    300..Connection
01d0  3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 43 6f   : keep-alive..Co
01e0  6f 6b 69 65 3a 20 53 49 44 3d 44 51 41 41 41 49   okie: SID=DQAAAI
01f0  55 41 41 41 43 54 6d 4d 38 74 36 55 74 54 6c 69   UAAACTmM8t6UtTli
0200  47 73 61 4c 58 61 75 61 36 39 5f 66 53 39 63 35   GsaLXaua69_fS9c5
0210  4f 6e 78 4e 77 6c 76 35 45 49 79 61 59 73 55 34   OnxNwlv5EIyaYsU4
0220  59 48 4c 6b 30 4d 5a 7a 72 76 66 63 33 31 79 64   YHLk0MZzrvfc31yd
0230  61 2d 32 53 73 5f 42 47 67 43 5f 4e 43 65 75 52   a-2Ss_BGgC_NCeuR
0240  55 38 43 73 69 6f 52 39 77 45 75 46 6c 47 43 5f   U8CsioR9wEuFlGC_
0250  37 77 70 4d 65 41 4a 49 4c 6f 66 79 44 4b 69 42   7wpMeAJILofyDKiB
0260  52 79 4b 4d 62 4b 73 4d 71 45 34 46 63 36 56 4c   RyKMbKsMqE4Fc6VL
0270  6c 48 50 79 32 38 67 79 4d 37 4a 42 44 67 34 51   lHPy28gyM7JBDg4Q
0280  57 50 41 33 49 51 46 57 4a 32 61 67 6d 74 45 33   WPA3IQFWJ2agmtE3
0290  7a 37 61 4b 61 39 77 64 48 43 4d 41 5f 34 70 68   z7aKa9wdHCMA_4ph
02a0  74 56 5a 6d 33 63 51 6b 71 2d 50 69 32 31 50 49   tVZm3cQkq-Pi21PI
02b0  32 63 54 5f 49 3b 20 50 52 45 46 3d 49 44 3d 64   2cT_I; PREF=ID=d
02c0  61 31 33 61 31 33 36 37 33 37 62 65 32 36 32 3a   a13a136737be262:
```

```
02d0  54 4d 3d 31 31 39 36 31 35 32 32 36 37 3a 4c 4d   TM=1196152267:LM
02e0  3d 31 32 30 31 38 35 32 37 37 32 3a 47 4d 3d 31   =1201852772:GM=1
02f0  3a 53 3d 38 35 47 5f 79 36 6a 45 48 6b 68 71 36   :S=85G_y6jEHkhq6
0300  45 36 47 0d 0a 43 61 63 68 65 2d 43 6f 6e 74 72   E6G..Cache-Contr
0310  6f 6c 3a 20 6d 61 78 2d 61 67 65 3d 30 0d 0a 0d   ol: max-age=0...
0320  0a                                                .

Frame 13
0000  00 17 42 6e da aa 00 1b d4 bd 6e d8 08 00 45 00   ..Bn......n...E.
0010  00 28 4e ed 00 00 30 06 59 eb d1 55 81 68 82 fb   .(N...0.Y..U.h..
0020  0c 3f 00 50 09 1d b1 ab 66 67 f3 e5 94 f7 50 10   .?.P....fg....P.
0030  1a 43 09 3c 00 00 00 00 00 00 00 00               .C.<........

Frame 14
0000  00 17 42 6e da aa 00 1b d4 bd 6e d8 08 00 45 00   ..Bn......n...E.
0010  01 c0 4e ee 00 00 30 06 58 52 d1 55 81 68 82 fb   ..N...0.XR.U.h..
0020  0c 3f 00 50 09 1d b1 ab 66 67 f3 e5 94 f7 50 18   .?.P....fg....P.
0030  1a 43 a2 69 00 00 48 54 54 50 2f 31 2e 31 20 33   .C.i..HTTP/1.1 3
0040  30 32 20 46 6f 75 6e 64 0d 0a 4c 6f 63 61 74 69   02 Found..Locati
0050  6f 6e 3a 20 68 74 74 70 3a 2f 2f 77 77 77 2e 67   on: http://www.g
0060  6f 6f 67 6c 65 2e 69 74 2f 0d 0a 43 61 63 68 65   oogle.it/..Cache
0070  2d 43 6f 6e 74 72 6f 6c 3a 20 70 72 69 76 61 74   -Control: privat
0080  65 0d 0a 44 61 74 65 3a 20 57 65 64 2c 20 32 31   e..Date: Wed, 21
0090  20 4d 61 79 20 32 30 30 38 20 31 36 3a 30 38 3a    May 2008 16:08:
00a0  35 31 20 47 4d 54 0d 0a 43 6f 6e 74 65 6e 74 2d   51 GMT..Content-
00b0  54 79 70 65 3a 20 74 65 78 74 2f 68 74 6d 6c 3b   Type: text/html;
00c0  20 63 68 61 72 73 65 74 3d 55 54 46 2d 38 0d 0a    charset=UTF-8..
00d0  53 65 72 76 65 72 3a 20 67 77 73 0d 0a 43 6f 6e   Server: gws..Con
00e0  74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 32 31 38   tent-Length: 218
00f0  0d 0a 0d 0a 3c 48 54 4d 4c 3e 3c 48 45 41 44 3e   ....<HTML><HEAD>
0100  3c 6d 65 74 61 20 68 74 74 70 2d 65 71 75 69 76   <meta http-equiv
0110  3d 22 63 6f 6e 74 65 6e 74 2d 74 79 70 65 22 20   ="content-type"
0120  63 6f 6e 74 65 6e 74 3d 22 74 65 78 74 2f 68 74   content="text/ht
0130  6d 6c 3b 63 68 61 72 73 65 74 3d 75 74 66 2d 38   ml;charset=utf-8
0140  22 3e 0a 3c 54 49 54 4c 45 3e 33 30 32 20 4d 6f   ">.<TITLE>302 Mo
0150  76 65 64 3c 2f 54 49 54 4c 45 3e 3c 2f 48 45 41   ved</TITLE></HEA
0160  44 3e 3c 42 4f 44 59 3e 0a 3c 48 31 3e 33 30 32   D><BODY>.<H1>302
0170  20 4d 6f 76 65 64 3c 2f 48 31 3e 0a 54 68 65 20    Moved</H1>.The
0180  64 6f 63 75 6d 65 6e 74 20 68 61 73 20 6d 6f 76   document has mov
0190  65 64 0a 3c 41 20 48 52 45 46 3d 22 68 74 74 70   ed.<A HREF="http
01a0  3a 2f 2f 77 77 77 2e 67 6f 6f 67 6c 65 2e 69 74   ://www.google.it
01b0  2f 22 3e 68 65 72 65 3c 2f 41 3e 2e 0d 0a 3c 2f   /">here</A>...</
01c0  42 4f 44 59 3e 3c 2f 48 54 4d 4c 3e 0d 0a         BODY></HTML>..

Frame 15
0000  00 30 48 8d ea ba 00 17 42 6e da aa 08 00 45 00   .0H.....Bn....E.
0010  00 3b 68 86 00 00 80 11 b3 d6 82 fb 0c 3f 82 fb   .;h..........?..
0020  0c 20 04 01 00 35 00 27 3d bb 75 b0 01 00 00 01   . ...5.'=.u.....
0030  00 00 00 00 00 00 03 77 77 77 06 67 6f 6f 67 6c   .......www.googl
0040  65 02 69 74 00 00 01 00 01                       e.it.....

Frame 16
0000  00 17 42 6e da aa 00 30 48 8d ea ba 08 00 45 00   ..Bn...0H.....E.
```

```
0010  01 0b 00 00 40 00 40 11 1b 8d 82 fb 0c 20 82 fb   ....@.@...... ..
0020  0c 3f 00 35 04 01 00 f7 61 1c 75 b0 81 80 00 01   .?.5....a.u.....
0030  00 05 00 07 00 00 03 77 77 77 06 67 6f 6f 67 6c   .......www.googl
0040  65 02 69 74 00 00 01 00 01 c0 0c 00 05 00 01 00   e.it............
0050  03 ab 28 00 10 03 77 77 77 06 67 6f 6f 67 6c 65   ..(...www.google
0060  03 63 6f 6d 00 c0 2b 00 05 00 01 00 02 41 e8 00   .com..+......A..
0070  08 03 77 77 77 01 6c c0 2f c0 47 00 01 00 01 00   ..www.l./.G.....
0080  00 01 28 00 04 d1 55 81 93 c0 47 00 01 00 01 00   ..(...U...G.....
0090  00 01 28 00 04 d1 55 81 68 c0 47 00 01 00 01 00   ..(...U.h.G.....
00a0  00 01 28 00 04 d1 55 81 63 c0 4b 00 02 00 01 00   ..(...U.c.K.....
00b0  00 eb 69 00 04 01 66 c0 4b c0 4b 00 02 00 01 00   ..i...f.K.K.....
00c0  00 eb 69 00 04 01 62 c0 4b c0 4b 00 02 00 01 00   ..i...b.K.K.....
00d0  00 eb 69 00 04 01 67 c0 4b c0 4b 00 02 00 01 00   ..i...g.K.K.....
00e0  00 eb 69 00 04 01 64 c0 4b c0 4b 00 02 00 01 00   ..i...d.K.K.....
00f0  00 eb 69 00 04 01 63 c0 4b c0 4b 00 02 00 01 00   ..i...c.K.K.....
0100  00 eb 69 00 04 01 61 c0 4b c0 4b 00 02 00 01 00   ..i...a.K.K.....
0110  00 eb 69 00 04 01 65 c0 4b                       ..i...e.K
```

……